We are IntechOpen,
the world's leading publisher of
Open Access books
Built by scientists, for scientists

6,900
Open access books available

185,000
International authors and editors

200M
Downloads

154
Countries delivered to

Our authors are among the
TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS
BOOK CITATION INDEX
INDEXED

WEB OF SCIENCE™
Selection of our books indexed in the Book Citation Index
in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com

The Effect of Packet Losses and Delay on TCP Traffic over Wireless Ad Hoc Networks

May Zin Oo and Mazliza Othman
*University of Malaya
Kuala Lumpur,
Malaysia*

1. Introduction

The popularity of wireless network has been growing steadily. Wireless ad hoc networks have been popular because they are very easy to implement without using base stations. The wireless ad hoc networks are complex distributed systems that consist of wireless mobile or static nodes that can freely and dynamically self-organize. The ad hoc networks allow nodes to seamlessly communicate in an area with no pre-existing infrastructure. Future advanced technology of ad hoc network will allow the forming of small ad hoc networks on campuses, during conferences and even in homes. Furthermore, there is an increasing need for easily portable ad hoc networks in rescue mission, especially for accessing rough terrains. However, the quick adaptation and ease of configuration of ad hoc networks come at a price.

In wireless ad hoc networks, route changes and network partitions occur frequently due to the unconstrained network topology changes. Moreover, this kind of network inherits the traditional problems of wireless communication, such as unprotected outside signals or interferences, unreliable wireless medium, asymmetric propagation properties of wireless channel, hidden and exposed terminal phenomena, transmission rate limitation and blindly invoking congestion control of transport layer. Although most of these limitations and complexities are due to the lack of fixed backbone or infrastructure, building ad hoc network temporarily is not only simple and easy to implement but also cost-effective and less time-consuming if compared to an infrastructure network that needs to establish a based station and fixed backbone. Among the above mentioned problems and limitations, the impact of transport layer limitations is analyzed across ad hoc routing protocols throughout the network topologies.

Transmission Control Protocol (TCP) (Postel, 1981) is the *de facto* standard designed to provide reliable end-to-end delivery of data packet in the wired networks. Normally, TCP is an independent protocol that is not related to the underlying network technology. However, some assumptions of TCP, such as consideration of only static node, packet losses due to congestion or buffer overflows are inspired from the features of wired networks. In the wireless network, these assumptions may not be correct all the time due to the rapid network topology changes, node movements and limited battery power. In order to apply TCP to an ad hoc environment, TCP has to overcome many problems, such as packet losses due to congestion, high bit errors, node mobility, longer delay and so on. The following TCP versions, Tahoe (Stevens, 1997), Reno (Allman, 1999), NewReno (Floyd & Henderson, 1999), Vegas (Brakno et al., 1994) and Westwood (Gerla et al., 2002), are enhanced versions of TCP and perform differently depending on how the routing protocols can quickly adapt route changes due to link breaks in an ad hoc network environment.

For wireless ad hoc networks, the issue of routing packets between any pair of nodes becomes a challenging task because the nodes can move randomly within the network. A path that is considered optimal at a given point in time might not work at all a few moments later. Traditional routing protocols such as DSDV (Perkins & Watson, 1994) are proactive in that they maintain routes to all nodes. They react to any change in the topology even if no traffic is affected by the change and they require periodic control messages to maintain routes to every node in the network. As mobility increases, more of scarce resources, such as bandwidth and power, will be used. Alternative reactive routing protocols, i.e. DSR (Johnson et al., 2007) and AODV (Perkins et al., 2003), determine the route when they explicitly need to route packets, thus avoiding nodes from updating every possible route in the network. However, these protocols tend to cause the broadcast storm problem (Tseng et al., 2002) due to the broadcast nature of the route discovery procedure. To avoid the discovery of a new route whenever a route fails, multipath routing protocols, i.e. AOMDV (Marina & Das, 2006) and OLSR (Clausen & Jacquet, 2003), were proposed which involve either on-demand or the usage of multiple relay points according to the link state information.

In the wireless ad hoc network, the behavior of protocols always vary depending on the core mechanisms of other protocols and factors such as node speeds, node movement patterns and background traffic. Almost all previous studies consider the importance of routing protocols over the performance of TCP (Ahuja at al., 2000; Dyer & Boppana, 2001; Gupta et al., 2004; El-Sayed, 2005; Kawadia & Kumar, 2005; Osipov & Tschudin, 2006; Mondal & Laqman, 2007; Anastasi et al., 2007; Sakib, 2009). Ahuja at al., (2000) considered four routing protocols: AODV, DSR, DSDV and SSA (Signal Stability-based Adaptive (Dube et al., 1997)) protocols and analyzed the performance of TCP. Dyer & Boppana (2001) also considered two on demand routing protocols, DSR and AODV, and proposed an adaptive proactive protocol (ADV) to enhance the TCP performance under a variety of conditions.

On the other hand, several papers (Ahuja at al., 2000; Chandran et al., 2001; Dyer & Boppana, 2001; Holland & Vaidya, 2002) discuss the effect of node mobility that may severely degrade the TCP performance due to the protocol's inability to manage efficiently mobility effects. As there are different versions of the TCP, many authors have compared the performance of different TCP versions by measuring throughput and fairness (Xu & Saadawi, 2000; Rakabawy et al. 2005, Kim et al., 2005). However, their analysis focus on the comparison of throughput and fairness, rarely considered packet loss rate depending on the increased number of connections. Some of them like, Kim et al. (2005), considered only TCP-NewReno and TCP-Vegas depending on AODV and OLSR routing protocols.

To the best of our knowledge, very few experimental analyses have been carried out so far (Lim et al., 2003; Oo & Othman, 2010) on the usage of multipath routing protocol. Their experiments are limited to using the ordinary TCP over multipath routing protocols. Therefore, this chapter discusses how the TCP variants interact to the use of routing protocols depending on the different topologies in the static and mobile ad hoc network environments.

The next section of this chapter is organized as follows. Section 2 briefly presents an overview of the ad hoc routing protocols and section 3 describes the variants of TCP that we have analyzed. Section 4 discusses the simulation methodology. Section 5 presents an analysis of the simulation results. Section 6 summarizes and concludes this chapter.

2. Overview of ad hoc routing protocols

2.1 Destination-Sequenced Distance Vector (DSDV)

DSDV (Perkins & Watson, 1994) is a proactive, hop-by-hop distance vector routing protocol. In DSDV, each node maintains a routing table of all possible destinations and the number of hops to each destination. Each node broadcasts its routing table information periodically throughout the network by using monotonically increased sequence numbers. The use of sequence number not only prevents the nodes from the occurence of stale routes but also avoids the formation of routing loops. If a node does not receive a periodic message from its neighbor for a while, it assumes that the link is broken. Moreover, its route update algorithm is very simple and guarantees loop free routes by transmitting a smaller update messages time to time. Therefore, the entire routing table need not be transmitted when the network topology changes occur.

2.2 Optimized Link State Routing Protocol (OLSR)

OLSR (Clausen & Jacquet, 2003) is a carefully designed protocol that works in a distributed manner and does not depend on any central entity. Each node chooses its neighbor nodes as multipoint relays (MPR) that are responsible for forwarding control traffic by flooding. MPRs provide the shortest path to a destination by declaring and exchanging the link information periodically for their MPR's selectors. By doing so, the nodes maintain the network topology information. The MPR is used to reduce the number of nodes that broadcasts the routing information throughout the network. To forward data traffic, a node selects its one hop symmetric neighbors, referred to as MPRset that covers all nodes that are two hops away.
The MPRset is calculated from information about the node's symmetric one hop and two hop neighbors. This information in turn is extracted from HELLO messages. Similar to the MPRset, a MPR Selectors set is maintained at each node. A MPR Selector set is the set of neighbors that have chosen the node as a MPR. Upon receiving a packet, a node checks its MPR Selector set to see if the sender has chosen the node as a MPR. If yes, the packet is forwarded, otherwise the packet is processed and discarded.
For route maintenance, Hello messages are broadcast periodically for link sensing, neighbor's detection and MPR selection process. The information contained in the HELLO message:

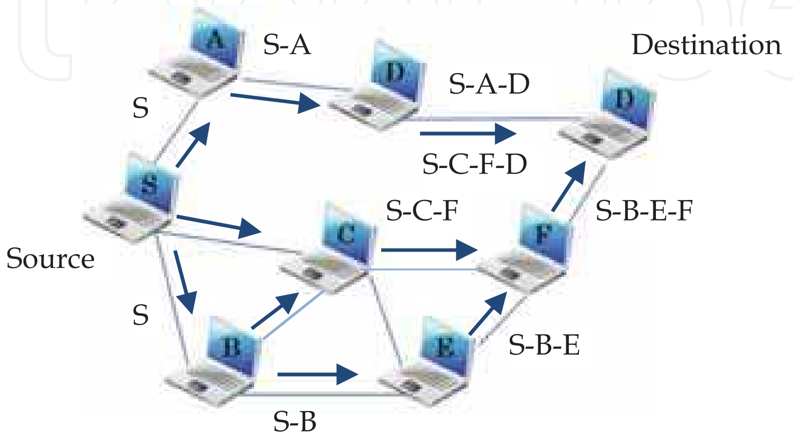
- how often the host sends Hello messages,
- willingness of a host to act as a Multipoint Relay, and
- information about its neighbor (i.e. interface address, link type and neighbor type)

The link type indicates that the link is symmetric, asymmetric or simply lost. The neighbor type is either symmetric, MPR or not a neighbor. If the link to the neighbor is symmetric, this node is chosen as MPR. After receiving a HELLO message information, a node builds its routing table. When a node receives a duplicate packet with the same sequence number, it discards the duplicate. A node updates its routing table either when a change in the neighbor is detected or a route to any destination has expired and a shorter route is detected for a destination.

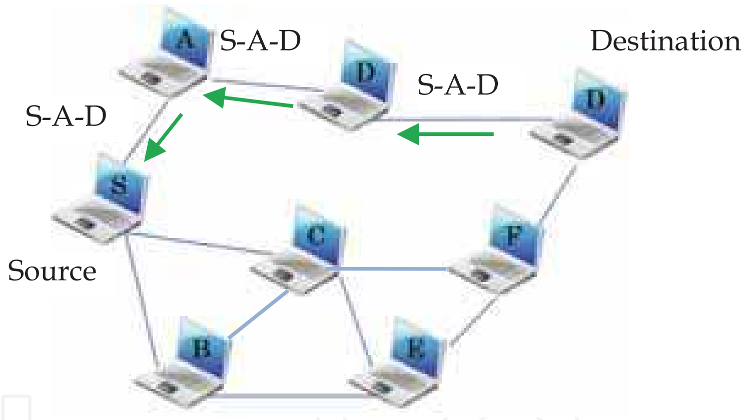
2.3 Dynamic Source Routing (DSR)

In DSR (Johnson et al., 2007), each node is initialized by broadcasting a route request packet when it either needs a route to the destination or does not have a route in its route cache. On

receiving this request, each node broadcasts it by appending its address to the request packet until this packet reaches the destination. The destination node replies to the earliest request to the source node. This approach is known as source routing.
In DSR, each node not only quickly supports a route when a route break occurs but also tolerates the topological changes due to the monitoring of the operations of routes. Moreover, it is able to compute the correct routes in the presence of asymmetric link. It does not make use of the periodic routing, thereby saving bandwidth and reducing power consumption.



(a) Sending procedure of a request packet

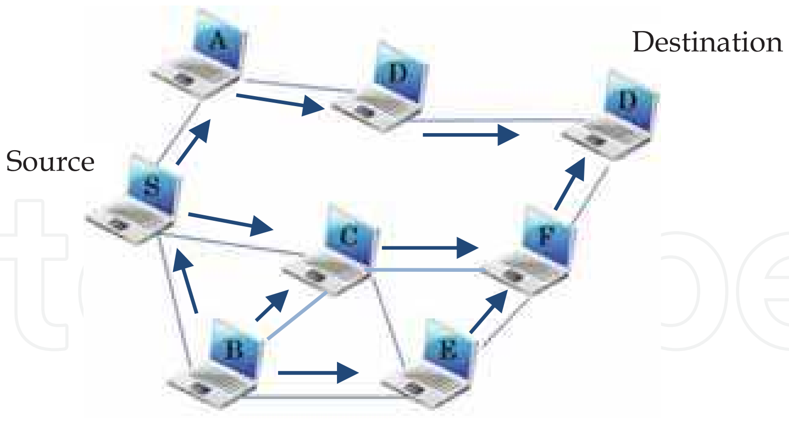


(b) Replying procedure of a reply packet

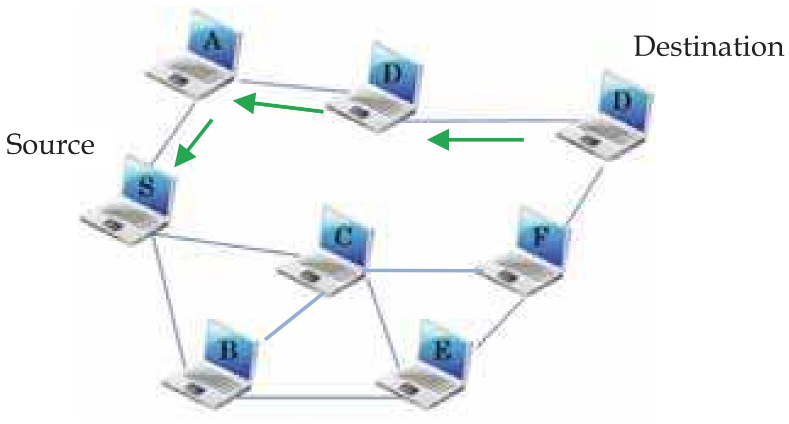
Fig. 1. Route discovery procedure of DSR

There are two main operations of DSR: route discovery and route maintenance. When a node wants to send a packet, and there is no route available to the destination, the node initiates a route discovery procedure. The source node broadcasts a route request to its neighbors by adding the destination address and route information that is recorded when the route request has passed. Upon receiving a route request, a node checks if it is the destination or if it knows a fresh route to the destination. If it is, the destination node has already found the complete route from the source and replied back to the source node. Otherwise, the node appends its address to the route information record and re-broadcasts the route request to its neighbors.To maintain the routes, each node constantly monitors the links it uses to forward the packets. If a node finds out that it cannot forward a packet, it sends a route error packet to its upstream nodes towards the source.

2.4 Ad-hoc On-demand Distance Vector (AODV)



(a) Sending procedure of a request packet



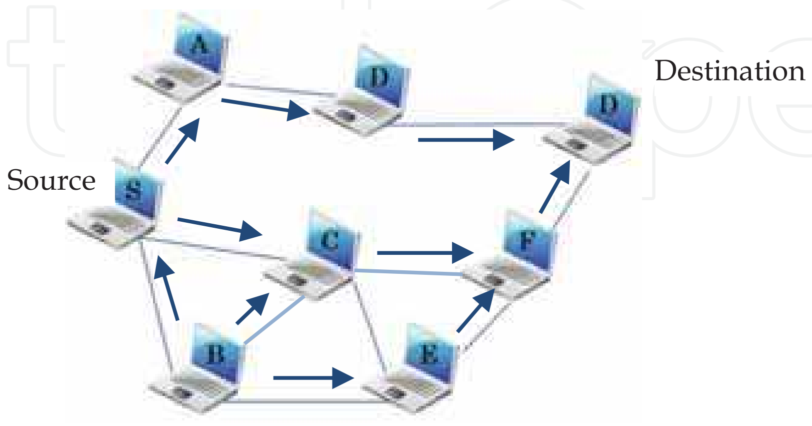
(b) Replying procedure of a reply packet

Fig. 2. Route discovery procedure of AODV

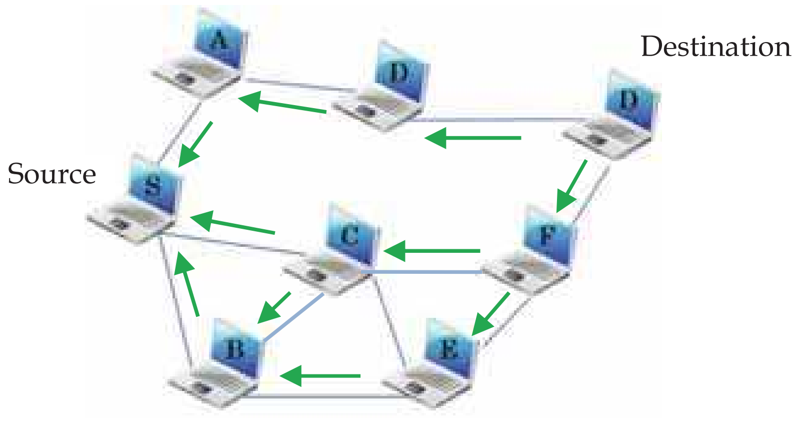
AODV (Perkins & Das, 2003) is based on DSDV and DSR routing protocols. In AODV, each node maintains a routing table, one entry per destination. Each entry records the next hop to the destination and its hop count (i.e. the distance from the current node to the destination node). AODV also uses a sequence number generated by a destination node to indicate the fresh-enough routes. Like DSR, AODV discovers a route through network-wide broadcasting. Unlike DSR, it does not record the nodes it has passed but only counts the number of hops. It builds the reversed routes to the source node by looking into the node that the route request has come. The responsibility of intermediate nodes is to check for fresh routes according to the hop count and destination sequence number and forwards the packets that they receive from their neighbors to the respective destinations.

AODV utilizes HELLO packets for route maintenance. If a node does not receive a HELLO packet within a certain time, or it receives a route break signal that is reported by the link layer, it sends a route error packet by either unicast or broadcast, depending on the precursor lists (i.e. active nodes towards the destination), in its routing table. It uses the periodic beaconing and sequence numbering procedures of DSDV and a similar route discovery procedure as in DSR. However, there are two major differences between DSR and AODV. The most distinguishing difference is that in DSR each packet carries full routing information, whereas in AODV the packets carry the destination address. This means that AODV is potentially less memory consuming than DSR. The other difference is that the route reply packets in DSR carry the address of every node along the route, whereas in AODV the route reply packets only carry only the destination IP address and sequence number. AODV avoids the stale route cache problem of DSR and it adapts the network topology changes quickly by resuming route discovery from the very beginning.

2.5 Ad-hoc On-demand Multipath Distance Vector (AOMDV)



(a) Sending procedure of a request packet



(b) Replying procedure of a reply packet

Fig. 3. Route discovery procedure of AOMDV

To overcome the invoking of a route discovery procedure whenever a route break occurs, Marina & Das (2006) proposed an AOMDV that allows each node to keep multiple paths to the destination. When a source node has data packets for a destination, it first checks its routing table to ascertain whether it already has a route to the destination node. If a route is available, it sends the data packets by utilizing its existing route. If not, it initiates a route discovery procedure by broadcasting RREQ to obtain a route to the intended destination.
AOMDV computes multiple paths and observes each route advertisement to define an alternate path to the source or the destination during a route discovery procedure. RREQ packets arriving at the nodes are copied and sent back to the source nodes. This approach may push the formation of loops due to accepting all copied routes. In order to eliminate any possibility of loops, it uses advertised hop count field in the route tables. The advertised hop count of a node S for a destination D is set the maximum hop count of the multiple paths for D at S.

The advertised hop count is initialized each time the sequence number is updated. By doing so, AOMDV only accepts alternative routes with lower hop counts. Each RREQ conveys an additional first hop field to indicate the first neighbor of the source node. The intermediate nodes do not discard duplicate copies of RREQ immediately as long as each RREQ provides a new node-disjoint path to the source. If an intermediate offers a new path, a reverse path is set up. It sends a RREP back to the source. At the destination, reverse routes are established like in the same situation of intermediate nodes. By computing multiple paths in a single route discovery attempt, a new route discovery is needed only when all paths fail.

3. Overview of transport protocols

3.1 TCP-Tahoe

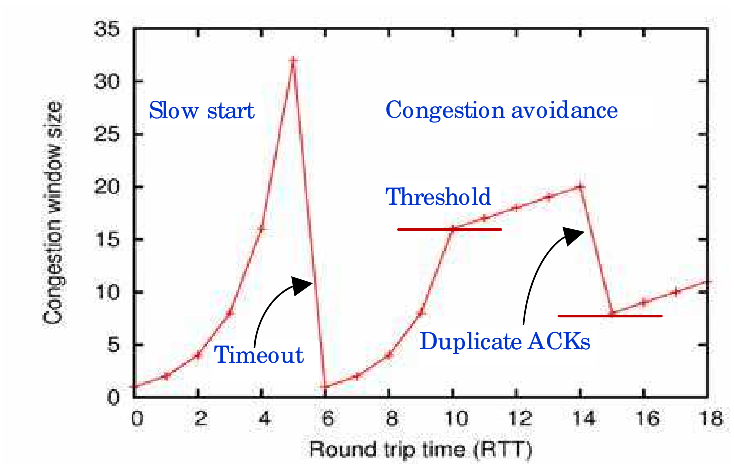


Fig. 1. Congestion control of TCP-Tahoe

The TCP protocol provides reliability, flow control, congestion avoidance, fairness, and in-order delivery. Originally, the protocol did not have congestion avoidance, causing the networks to become overloaded. TCP Tahoe introduced congestion avoidance, where dropped packets are used as an indication of congestion, and slow start, where the initial window size grows exponentially (i.e. a source node transmits one segment and wait for its ACK (acknowledgement). If the ACK is received, the congestion window is increased to transmit two segments. After receiving ACKs for those two segments, the congestion window is increased to four to transmit four segments) until either a congestion or timeout event is detected. In the congestion avoidance region, the initial window is increased linearly as shown in Fig. 1. In TCP-Tahoe (Stevens, 1997), there are two indications of packet losses: a timeout event and the receipt of duplicate ACKs. Whenever the timeout event occurs, Tahoe starts the slow start procedure by initiating congestion window size starting from one, whereas the congestion window (*cwnd*) is halved (i.e. *cwnd* = *cwnd*/2) when three duplicate ACKs are received.

3.2 TCP-Reno

Instead of starting transmission from a slow start after a relatively long idle period, Allman (1999) introduced TCP-Reno by adding fast retransmit and fast recovery algorithms. With fast retransmit, Reno attempts to retransmit packets before a timeout. However, a sender will initiate a slow-start procedure as if a timeout causes the retransmission. With fast recovery, Reno uses additive increase/ multiplicative decrease at all the time, and only initiates the slow start when either a connection is established or a timeout occurs. In other words, Reno with fast recovery omits the slow start if no timeout occurs.

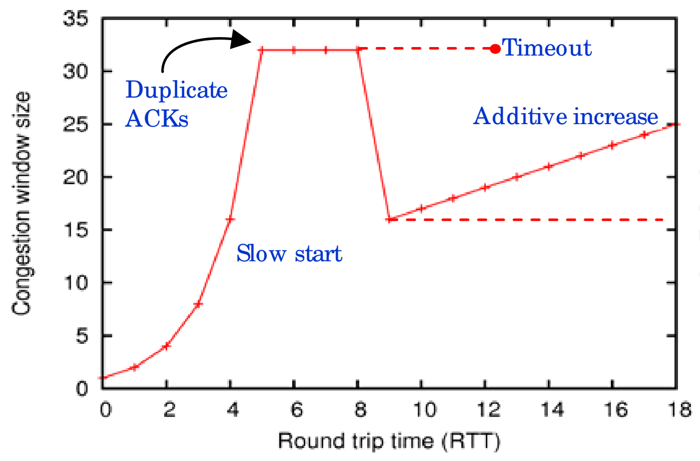


Fig. 2. Congestion control of TCP-Reno

3.3 TCP-NewReno

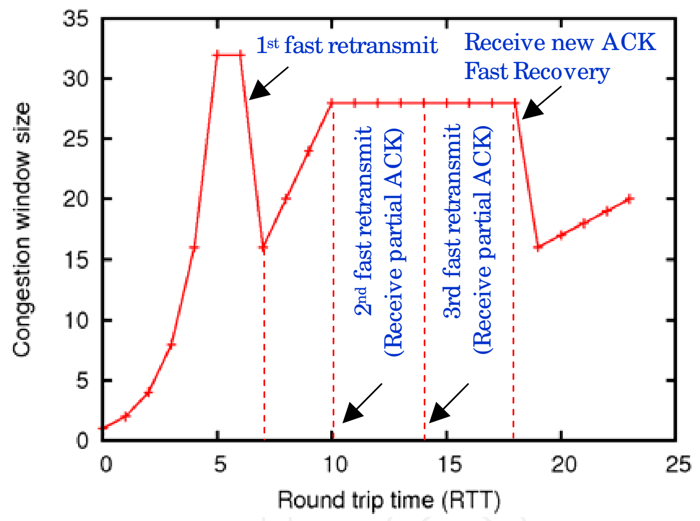


Fig. 3. Congestion control of TCP-NewReno

TCP-NewReno (Floyd & Henderson, 1999) is an improvement of Reno and it is an advanced fast transmit, where three duplicate acknowledgments signal a retransmission without a timeout with fast recovery. The fast recovery means that once a certain threshold of ACKs is received, the window size is decreased by half rather than starting over with slow start. Only during timeout does it go back into slowstart. NewReno increases the adoption of the TCP selective acknowledgements (SACK) (Mathis & Mahdavi, 1996) modification. TCP-NewReno possesses two kinds of ACKs: partial ACK and full ACK. The partial ACK acknowledges some segments at the fast recovery stage while the full ACK acknowledges all outstanding data. NewReno retransmits the segment based on the partial ACK. Upon receiving the full ACK, the sender sets the congestion window to slow start threshold and terminates the fast recovery. Then the congestion avoidance mechanism is resumed. In this way, the NewReno maintains a high throughput.

3.4 TCP-Vegas

Tahoe, Reno and NewReno variants are window-based transport protocols that adjust congestion window upon packet losses. On the other hand, Brakno et al., (1994) introduces a delay-based TCP, called TCP-Vegas, which does not violate the congestion avoidance paradigm of TCP. Instead of increasing the sending rate until a packet loss occurs, TCP Vegas prevents such losses by decreasing the sending rate when it senses incipient congestion even if there is no indication of packet loss. Vegas uses packet delay as an indication of congestion.
In a situation when a duplicate ACK is received, the timestamp for the ACK is compared to a timeout value. If the timestamp is greater than the timeout value, then Vegas will retransmit rather than wait for three duplicate ACKs. Vegas detects congestion at an incipient stage based on increasing Round Trip Time (RTT) values of the packets in the connection unlike other flavors, like NewReno, which detect a congestion only after it has actually happened via packet drops. TCP Vegas adopts a more sophisticated bandwidth estimation scheme. It uses the difference between expected and actual flow rates to estimate the available bandwidth in the network. When the network is not congested, the actual flow rate will be close to the expected flow rate. Otherwise, the actual flow rate will be smaller than the expected flow rate. So, TCP-Vegas can estimate the congestion level in the network and updates the window size accordingly. The difference between the flow rates can be easily calculated during the round trip time using the equation

$$Diff = (Expected \mid Actual)\, BaseRTT$$

where *Expected* is the expected rate, *Actual* is the actual rate, and *BaseRTT* is the minimum round trip time. Based on *Diff*, the source updates its window size as follows.

$$CWND = \begin{cases} CWND + 1 & if\ Diff < \alpha \\ CWND - 1 & if\ Diff > \beta \\ CWND & otherwise \end{cases}$$

3.5 TCP-Westwood

TCP-Westwood (Gerla et al., 2002) is a sender-side modification of the TCP congestion window algorithm. The key idea behind it is to estimate bandwidth to control the congestion window and the slow start threshold by monitoring the ACK packets.
A sender measures the rate of ACKs that it receives and estimates the data rate currently achieved by that connection. Whenever the packet losses occur (i.e. timeout or duplicate ACKs), the sender estimates the bandwidth to properly set the congestion window and slow start threshold. Instead of halving congestion window like Reno and NewReno, TCP Westwood backs off some value of cwnd and threshold based on the estimated value to ensure faster recovery. The improvement of Westwood is more significant in wireless networks with lossy links, since TCP Westwood relies on end-to-end bandwidth estimation to discriminate the cause of packet loss. Rather, it fully complies with the end-to-end TCP design principle.

4. Simulation methodology

We use simulations to study the variants of TCP over three ad hoc routing protocols. The simulation study is done using Network Simulator (NS-2) (McCanne & Floyd). NS-2 is a

discrete event simulator that was developed as part of the VINT project at the Lawrence Berkeley National University.
For the performance of TCP variants over routing protocols in a static environment we simulate a scenario of chain (6 nodes) and grid (25 nodes) in a rectangular topology of 1300m × 1000m, where each node has a transmission range of 200m. All nodes have a default bandwidth of 11Mbps and the simulation period is 360 seconds. We use an FTP (File Transfer Protocol) application with a packet size of 512 bytes. Each TCP variant is run over each routing protocol in static and mobile environments (Figs. 4 to 7).

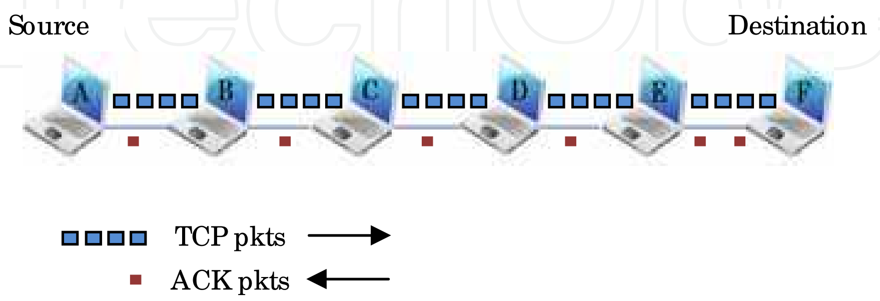


Fig. 4. Source node A connects to destination node F in a static ad hoc network

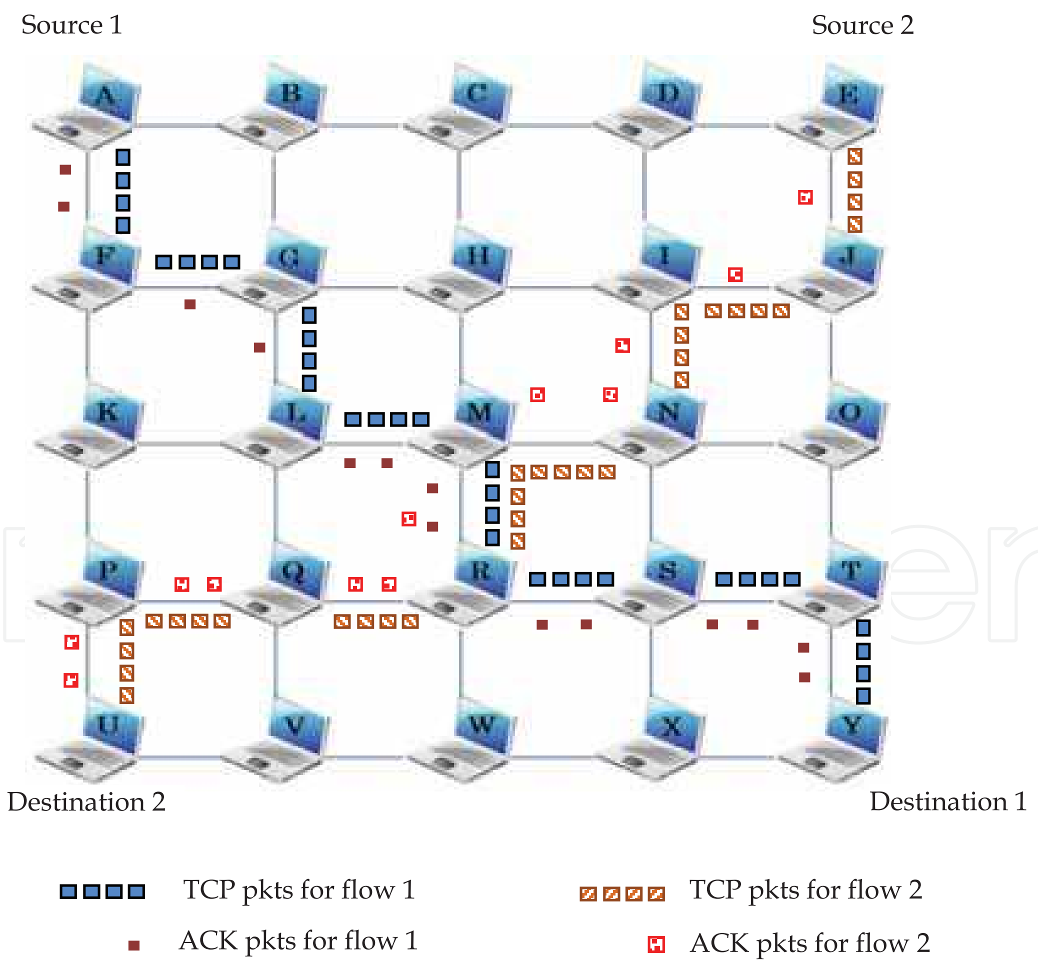


Fig. 5. Two TCP connections between a pair of nodes (A and Y) and node (E and U) in a static ad hoc network

In a mobile ad hoc environment, we manually change the network topology by adding movement for a few nodes. The "*setdest*" command under NS-2 directory (i.e. *ns-allinone-2.34\ns-2.34\indep-utils\cmu-scen-gen\setdest*) is used to generate the node movement. In Fig. 6, node C and node D start moving at 50 and 100 seconds of simulation time respectively. Both nodes turn back to its original position at 250 seconds, and they move at 10 m/ s speed. Similarly, in Fig. 7, node I and node J start moving at 20 seconds and return to

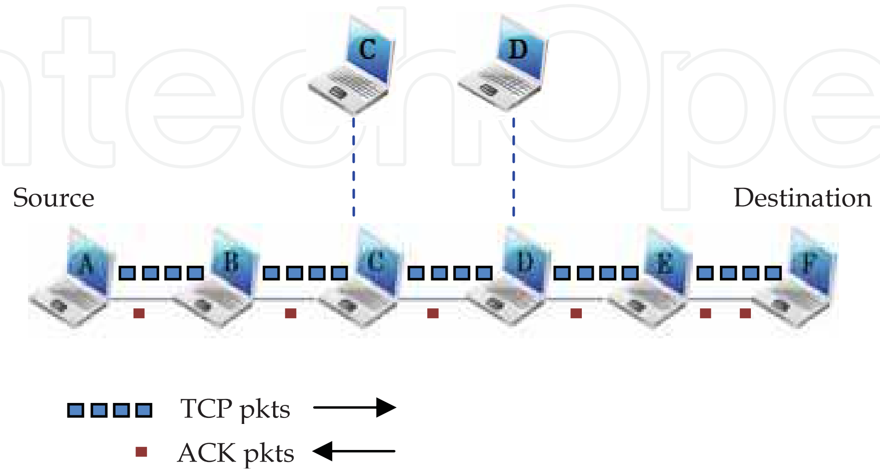


Fig. 6. Source node A connects to destination node F in mobile ad hoc network

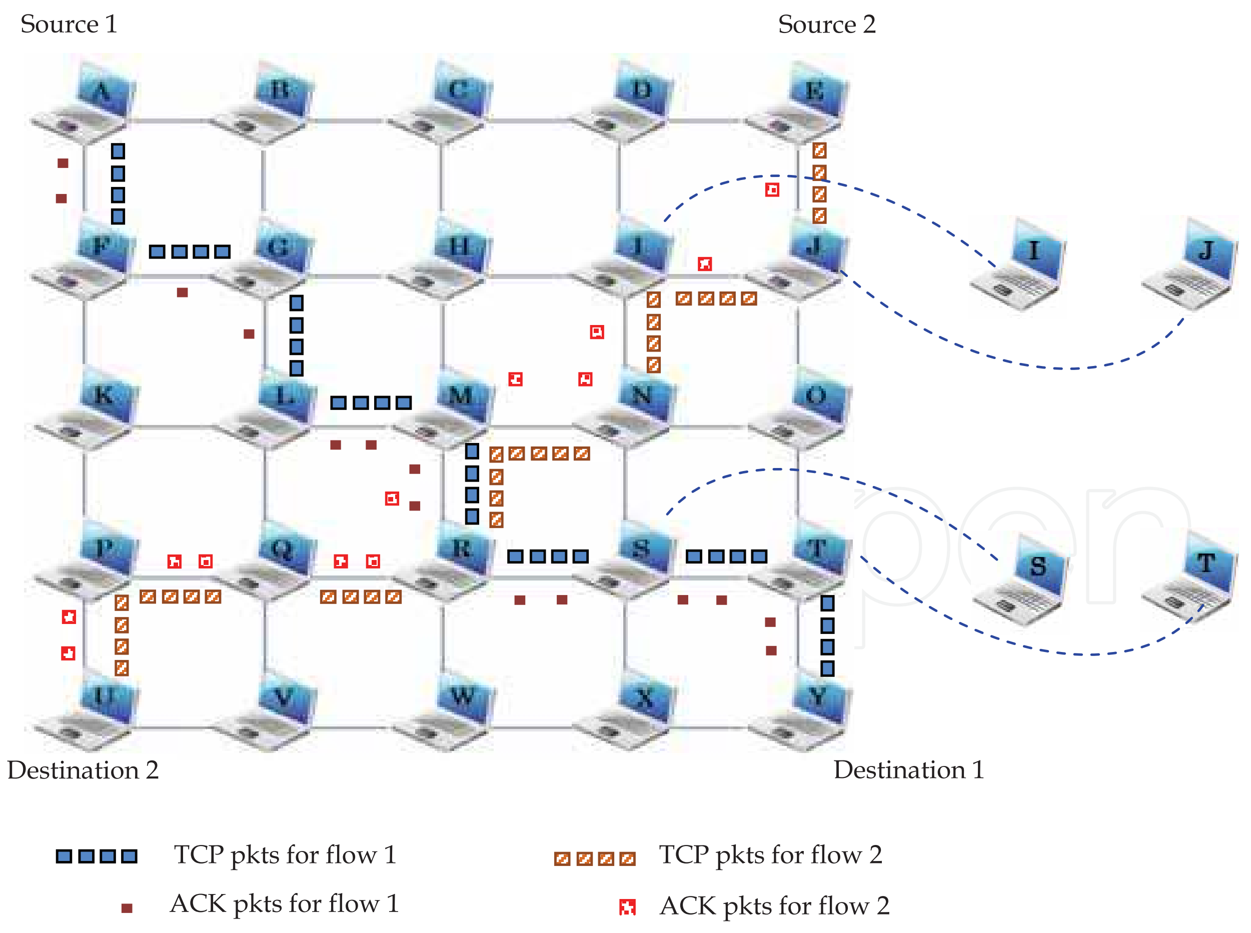


Fig. 7. Two TCP connections between a pair of nodes (A and Y) and node (E and U) in mobile ad hoc environment

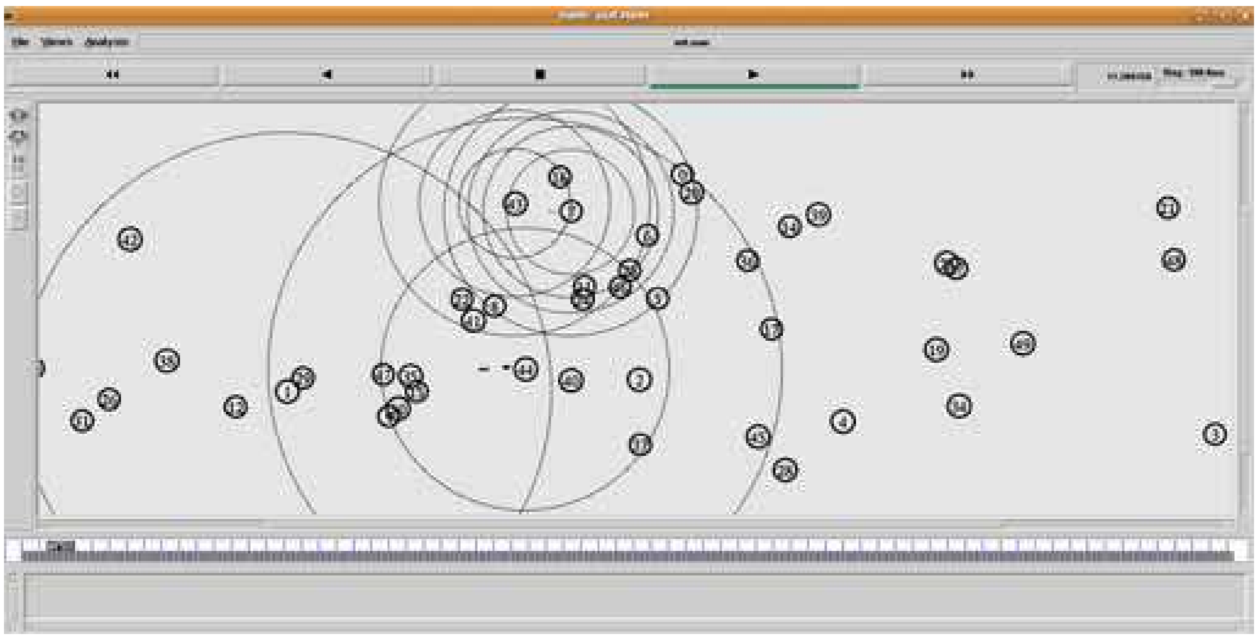


Fig. 8. The 50 pairs of random node movement in mobile environment

their original positions at 260 seconds. Also, node S and node T move to the left at 100 seconds and reach to their original positions at 300 seconds. In a chain topology, only one TCP connection is exchanged between a pair of source and destination while two TCP connections are transmitting in the grid topology for the static and mobile ad hoc environments.

To examine the TCP performance in a random topology, a moderate network of 50 nodes are randomly moved using Random Waypoint (RWP) mobility model (Camp et al., 2002) which is generated using the Bonnmotion v1.4 tool developed by the Communication Systems group at the Institute of Computer Science IV of the University of Bonn, Germany (*BonnMotion: a Mobility Scenario Generation and Analysis Tool 2009*). For example, the RWP mobility model can be generated by using the following command.

```
bm –f scenario1 RandomWaypoint –n 50 –d 900 –i 3600 –x 1600 –y 400 –h 10 –l 10 –p 0
```

n: the number of nodes that we wish to set
d: the simulation time
i: the cutting value that must be high default value because nodes have a higher probability of being near the center of the simulation area, while they are initially uniformly distributed over the simulation area in Random Waypoint model.
x: the coordinate of node position in x axis
y: the coordinate of node position in y axis
h: the maximum node speed
l: the minimum node speed
p: the maximum pause time

In the analyzed scenarios, the maximum pause time is set to zero for continuous movement, and nodes are allowed to move at 10 m/s speed. Simulations are run for 360 seconds simulation time. The number of tcp connections is varied between 10 to 50, and our performance analysis is examined by measuring the packet loss rate, delay and throughput.

5. Simulation results

In this section, we describe the results obtained from the simulation experiments in different scenarios. We simulate each variant of TCP (i.e. Tahoe, Reno, NewReno, Vegas and Westwood) over each routing protocol (i.e. DSDV, DSR, AODV, AOMDV and OLSR) in static and mobile ad hoc network environments. Then we measure how the topology changes affect the performance of TCP variants across each routing protocol in a 6-node chain and 25-node (5 x 5) grid topologies.
To examine the performance in the scenarios of random movement, the 50 pairs of nodes are simulated in 1600 x 400 simulation area for 360 seconds.
To analyze the network performance in those topologies, the packet loss rate (%), average end-to-end delay (msec) and throughput (kbps) are measured as performance metrics.
The packet loss rate (%) is the number of packet losses at the application layer while transferring data packets, i.e.

$$PLR = \frac{Dropped\ Packets}{Highest\ Packet\ ID + 1} * 100$$

The average end-to-end (EtE) delay (msec) is the transmission delay of data packets that are delivered to the intended destination successfully.
The throughput (kbps) is the rate of successfully delivered data per second to individual destinations during the network simulation.

5.1 Chain topology

Firstly, we analyze the packet loss rates of the TCP variants over the ad hoc routing protocols in static and mobile environments. The TCP variants over AODV incurs a lower packet loss rate than DSDV and DSR in static environment as shown in Table 1. Because DSDV sends periodic messages throughout the network, and DSR stores all route information in the control packets, the packet loss rates for both protocols increase due to the collision and congestion in the MAC layer. However, when the node movements are

Packet loss rate (%) / TCP variants	DSDV		DSR		AODV	
	Static	Mobile	Static	Mobile	Static	Mobile
Tahoe	0.09	0.75	0.09	0.14	0.05	0.45
Reno	0.09	0.75	0.09	0.14	0.05	0.45
NewReno	0.09	0.75	0.09	0.14	0.05	0.45
Vegas	0.00	0.46	0.00	0.11	0.00	0.33
Westwood	0.09	0.65	0.09	0.14	0.05	0.45

Table 1. The percentage of packet loss rate in chain topology

added, DSR achieves the lowest loss rate due to its route cache mechanism where all possible routes to the destination are kept. On the other hand, among the TCP variants, Vegas is the best protocol for all situations, and it achieves no losses over the routing protocols in the static environment.

Secondly, when we examine the average end-to-end delay, AODV incurs the lowest delay, whereas DSR incurs the highest delay over all TCP variants as shown in Table 2. AODV always keeps routes as a soft state, for example, routes expire after a timeout interval and a fresh route discovery is invoked. Accordingly, AODV is significantly better delay-wise and can possibly perform even better than others when node movements are added. Likewise, AODV has a special timer mechanism to detect route breaks and update fresh-enough routes whereas DSR does not contain any explicit mechanism to expire stale routes in the cache. The stale routes are later detected by route error packets, leading to performance degradation although it achieves a lower packet loss rate as shown in Table 1.

Delay (msec) / TCP variants	DSDV		DSR		AODV	
	Static	Mobile	Static	Mobile	Static	Mobile
Tahoe	432.4	604.7	646.6	639.7	418.5	415.0
Reno	432.4	604.7	646.6	639.7	418.5	415.0
NewReno	432.4	604.7	646.6	639.7	418.5	415.0
Vegas	72.9	333.4	71.3	70.5	67.79	67.4
Westwood	432.4	624.5	646.6	639.7	418.5	415.0

Table 2. The average end-to-end delay in chain topology

The delay-based Vegas achieves the lowest delay for the static and mobile ad hoc environments. The performance of Tahoe, Reno, NewReno and Westwood are not different enough to compare against each other in both environments. The delay difference of Vegas is lower than others by a factor of around 6 over DSDV and AODV, and by a factor of more than 9 over DSR for the static environment. However, the variants of TCP incur a lower delay over DSR and AODV whereas DSDV incurs a higher delay when the node movements are added. Especially, the delay of Vegas over DSDV suddenly increases once the nodes move as shown in Fig.1. No matter what variants of TCP are utilized, all of them achieve a lower delay over AODV routing protocol in both environments.

Throughput (kbps) / TCP variants	DSDV		DSR		AODV	
	Static	Mobile	Static	Mobile	Static	Mobile
Tahoe	97.5	27.8	97.4	39.9	101.4	30.6
Reno	97.5	27.8	97.4	39.9	101.4	30.6
NewReno	97.5	27.8	97.4	39.9	101.4	30.6
Vegas	74.4	17.5	100.4	33.9	104.6	39.8
Westwood	97.5	25.9	97.4	39.9	101.4	30.6

Table 3. The performance throughput in chain topology

Finally, Table 3 compares the throughput of TCP variants over each routing protocol. AODV supports higher throughput for all TCP variants, especially Vegas in both environments. However, the performance of Vegas is lower than others over DSDV in both environments. In DSR, Vegas achieves higher throughput than others in static environment, whereas its performance is lower than others in mobile environment. When the node movement is introduced, TCP variants over DSR achieve a higher throughput than DSDV and AODV because nodes with DSR always have backup routes in hand and keep them in their caches. As soon as a route break occurs due to congestion or collision, it can recover the route quickly before the TCP timeout. In this way, DSR attains higher throughput at moderate node movement in mobile environment as long as its route cache is not stale.

5.2 Grid topology

For the grid topology, one of the multipath routing protocols, AOMDV is considered to examine the performance of TCP variants. For the static environment, we encounter that TCP variants except Westwood have no packet loss over AODV as shown in Table 4. Another thing is that AOMDV also achieves a lower packet loss rate if compared to DSDV and DSR. However, finding multiple paths in a static environment is not effective if compared to the single path AODV, and even when a few node movement is added, AODV has a lower losses than AOMDV. AOMDV possibly performs better than AODV over the lossy links that occur due to the random node movement and increased traffic because it has fresh multiple alternative routes. Like in chain topology, Vegas upholds a lower packet loss than others, and there are no losses except over DSDV in the static environment.

Packet loss rate (%) / **TCP variants**	**DSDV**		**DSR**		**AODV**		**AOMDV**	
	Static	Mobile	Static	Mobile	Static	Mobile	Static	Mobile
Tahoe	0.29	1.73	0.15	0.88	0.00	0.33	0.06	0.57
Reno	0.29	1.73	0.15	0.86	0.00	0.33	0.06	0.57
NewReno	0.45	1.89	0.15	0.44	0.00	0.33	0.06	0.82
Vegas	0.17	0.20	0.00	0.14	0.00	0.07	0.00	0.18
Westwood	0.91	1.81	0.13	0.12	0.04	0.51	0.06	0.82

Table 4. The percentage of packet loss rate in grid topology

In Table 5, if we look at the end-to-end delay for all TCP variants in a static environment, DSDV has the lowest delay for both environments in the grid topology because the nodes in the grid topology are organized, therefore packet losses due to route break, congestion or collision of MAC layer could be recovered easily. The table-driven and periodic approach of DSDV, thus, suffers more losses possibly due to congestion, whereas it achieves the lowest delay compared to others. The delay of Westwood is the worst over DSR routing protocol in the static environment.

The delay-based protocol, Vegas always incurs a lower delay for all situations due to the consideration of actual and expected flow rates. On the other hand, Vegas obtains a lower delay over DSDV and DSR, and delay becomes higher over AODV and AOMDV protocols when the node movements are added. Although AODV achieves a lower delay in most situations, in the grid topology, it suffers a higher delay than others because the number of route discovery frequencies of AODV increases due to its flooding nature whenever a route break occurs due to network congestion.

Delay (msec) / TCP variants	DSDV		DSR		AODV		AOMDV	
	Static	Mobile	Static	Mobile	Static	Mobile	Static	Mobile
Tahoe	596.8	131.2	970.3	198.8	662.1	647.1	715.7	616.1
Reno	588.9	141.0	970.3	193.3	662.1	647.1	715.7	616.1
NewReno	614.5	134.7	970.3	183.3	662.1	647.1	715.7	662.1
Vegas	137.2	48.1	112.8	32.0	108.4	127.9	115.5	125.9
Westwood	574.4	150.6	1160.9	210.4	661.3	639.6	715.7	662.1

Table 5. The average end-to-end delay in grid topology

In Table 6, when throughput is compared, TCP variants over DSR perform better than others in both environments. In DSDV and AODV, Westwood is the best throughput in static environment, whereas it suffers the lowest throughput in mobile environment. In the grid topology, the possibility of congestion increases due to the channel contention. Whenever a packet loss occurs, Westwood attempts to select a slow start threshold and a congestion window depending on the effective bandwidth used at the time congestion is experienced, whereas Reno and NewReno blindly halves the congestion window after trying the fast retransmit and fast recovery procedures. Therefore, in grid topology, the performance of Westwood is significant if compared to others. On the other hand, the significance of performance throughput for all TCP variants can be seen over DSR routing in the mobile environment. DSR's route cache mechanism may not be effective enough to provide the routes that have been cached in high mobility and traffic scenarios, whereas in moderate situation, such as fewer node movement, DSR provides the highest throughput to all TCP variants.

Throughput (kbps) / TCP variants	DSDV		DSR		AODV		AOMDV	
	Static	Mobile	Static	Mobile	Static	Mobile	Static	Mobile
Tahoe	37.4	78.1	90.5	237.1	45.3	44.4	42.4	80.3
Reno	36.2	78.1	90.5	147.9	45.3	44.4	42.4	80.3
NewReno	37.4	76.4	90.5	161.1	45.3	44.4	42.4	39.2
Vegas	27.7	56.8	44.5	158.7	46.5	44.1	43.4	42.6
Westwood	56.2	36.2	60.7	123.4	54.4	52.9	42.4	39.2

Table 6. The performance throughput in grid topology

5.3 Random topology

As mentioned in section 4, we examine the performance of TCP variants and routing protocols in the random topology. All nodes move randomly across the RWP model. A node starts moving from a randomly chosen position and stays in one location for a certain period of time (i.e. a pause time). Once this time expires, the node chooses a destination and moves at a randomly chosen speed. This speed is selected from a uniformly distributed speed between minimum and maximum speed. Upon arrival at the destination, the above process is started over again.

5.3.1 Packet loss rate measurement

We vary the number of TCP connections from 10 to 50 connections and the network performance are measured in the 50 nodes random topology. TCP traffic is generated using a traffic generation tool under the NS-2 directory (i.e. *ns-allinone-2.34\ns-2.34\indep-utils\cmu-scen-gen*), for example, (*ns cbrgen.tcl -type <cbr|tcp> -nn <nodes> -seed <seed> -mc <connections> -rate <rate>)*. For the 20 number of connections and the 50 pair of nodes, the following command is used.

ns cbrgen.tcl –type cbr –nn 50 –seed 1 –mc 20 –rate 4

The percentage of packet loss rates for all TCP variants varies between 0.5 and 4 over all ad hoc routing protocols as shown in Fig. 9. The stability of TCP variants is encountered over DSR, and all losses vary between 0.5 and 1.2 (Fig. 9(b)). As the number of TCP connections increases, the packet loss rates of TCP variants decrease. In Fig. 9(c), the packet losses over AODV are the worst if we also look at the view of stability issue. The link failure detection mechanism of AODV based on HELLO messages generates frequent route failures with associated packet loss oscillation.

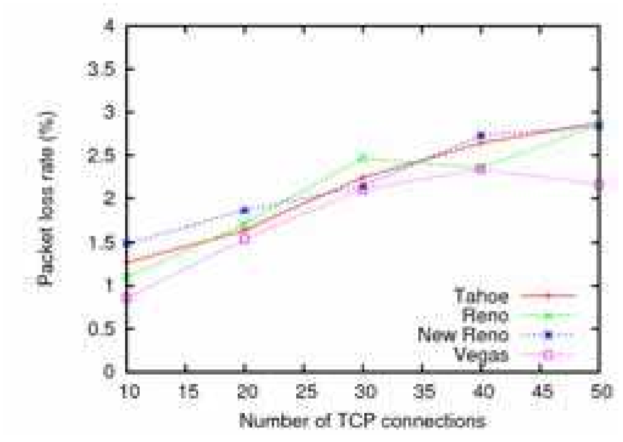
In the two multipath routing protocols AOMDV and OLSR, AOMDV encounters a greater packet loss rate than OLSR by a factor of up to 2 as shown in Fig. 9(d) and (e). Although AOMDV supports multiple paths between a source and destination, it is difficult to recover the packets during the time between the failure of a primary route and the finding of an alternative route. On the other hand, as OLSR nodes always have routes in hand due to its proactive nature, it reduces packet loss rates significantly (Oo & Othman, 2010).

Vegas is the best transport variant of TCP, and it is able to provide a lower packet loss rate in most situations. It is able to detect congestion in advance by estimating bandwidth before actual congestion happens. Other TCP variants like Tahoe, Reno, NewReno are not as good as Vegas when TCP connection flows grow, especially in DSDV and AODV.

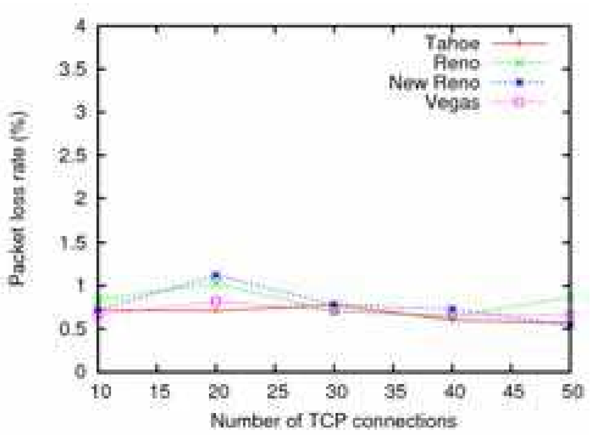
5.3.2 Average end-to-end delay measurement

Although DSR has the lowest packet loss rate as mentioned in section 5.3.1, it is not good enough to apply in delay-sensitive applications. It suffers the highest delay especially for Tahoe, Reno and NewReno as shown in Fig. 10 (a). As the number of TCP connections increases, the average delay of TCP variants also increases. Vegas can transfer packets almost four times faster than over DSDV and OLSR, two times over AODV and DSR, five times over AOMDV than others as shown in Fig. 10.

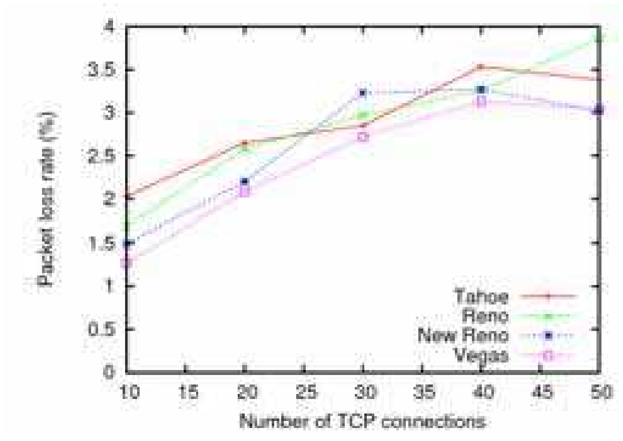
The performances of window-based protocols such as Tahoe, Reno and NewReno are not very significant over each other, whereas delay-based Vegas protocol gains a significantly lower delay. On the other hand, when the route breaks occur, Tahoe, Reno and NewReno halves it congestion window and starts the slow start procedure after the TCP timeout expiry period, tending to the increased delay if compared to the Vegas. In Fig. 10 (a), TCP variants over DSDV are the best if compared to other routing protocols. DSDV starts discovering routes proactively, and it may increase the routing overhead, whereas it significantly reduces average end-to-end delay at the moderate network.



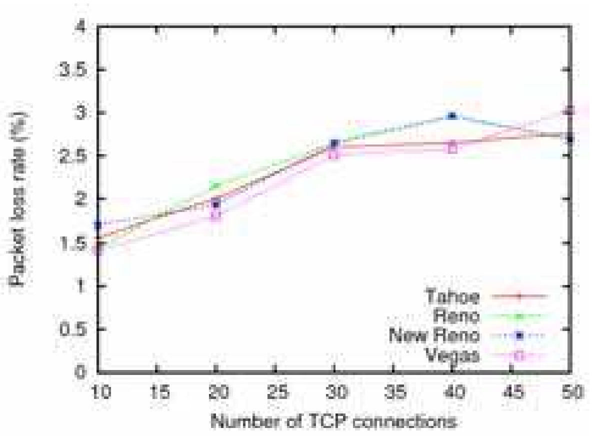
(a) DSDV



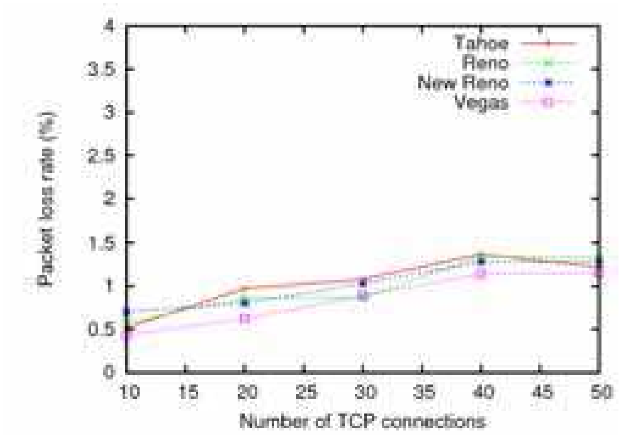
(b) DSR



(c) AODV

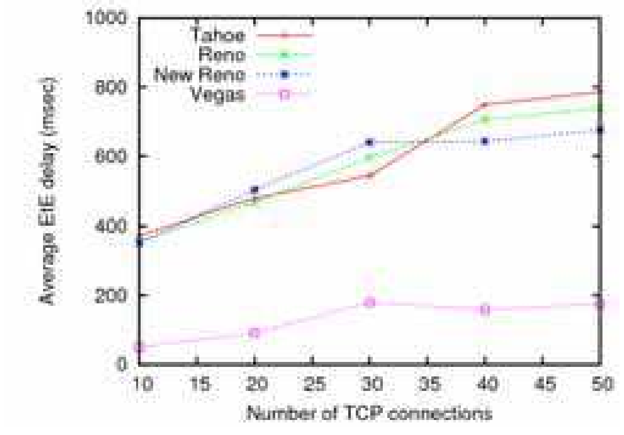


(d) AOMDV

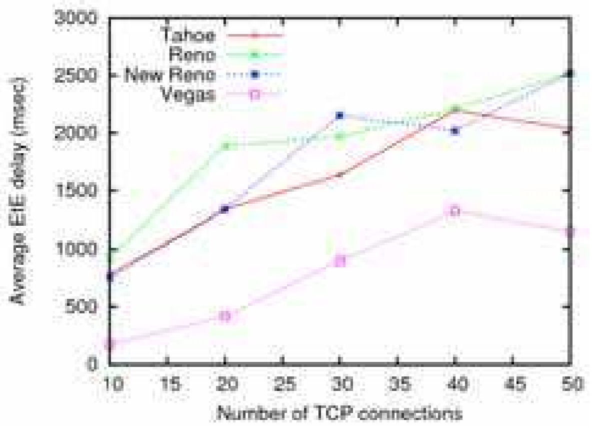


(e) OLSR

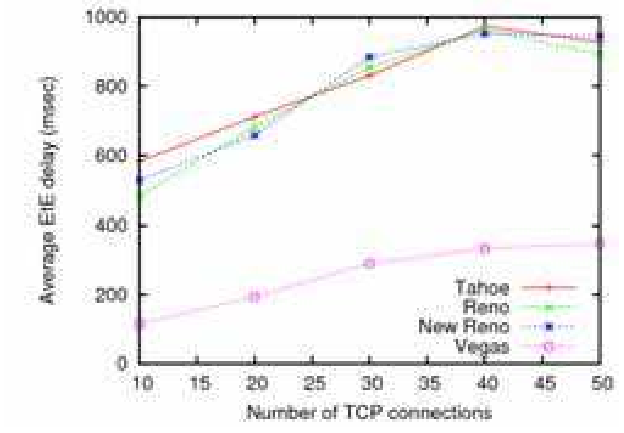
Fig. 9. Packet loss rates measurement in the random topology



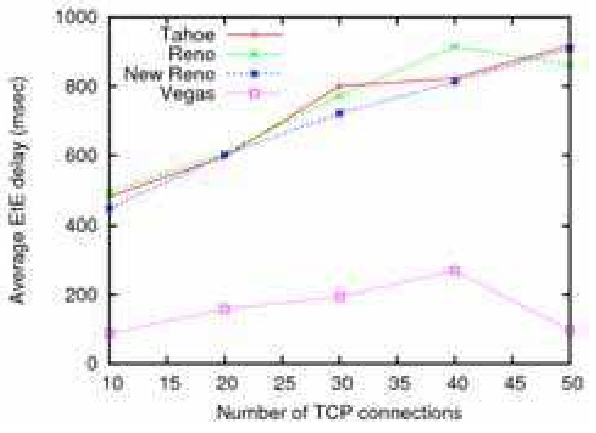
(a) DSDV



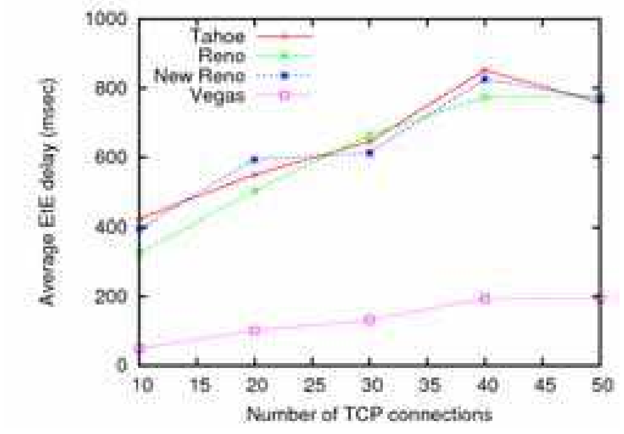
(b) DSR



(c) AODV

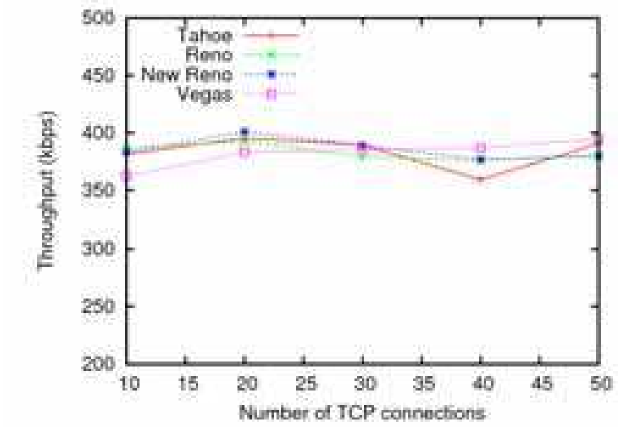


(d) AOMDV

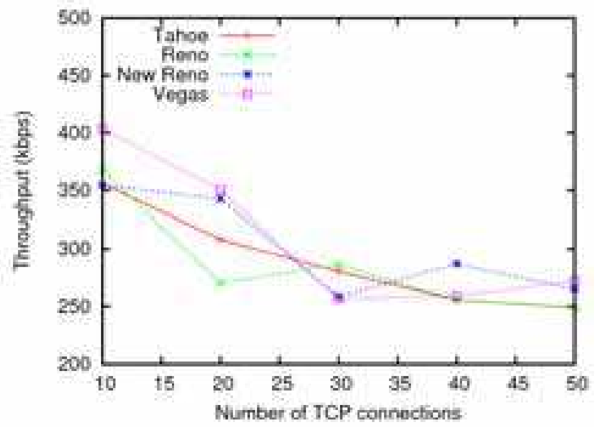


(e) OLSR

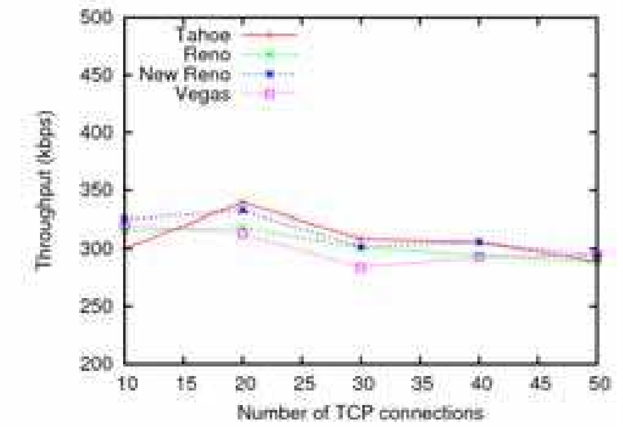
Fig. 10. Average end-to-end delay measurement in the random topology



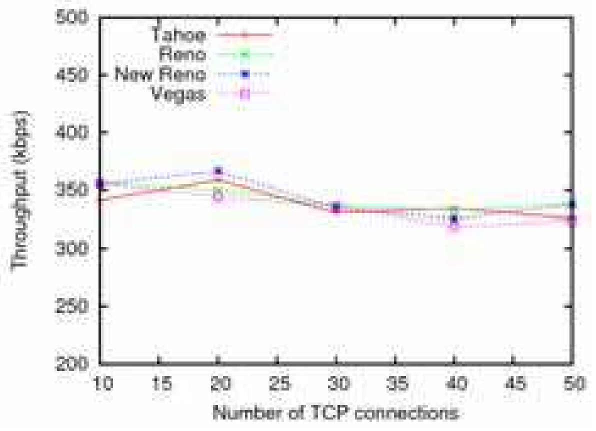
(a) DSDV



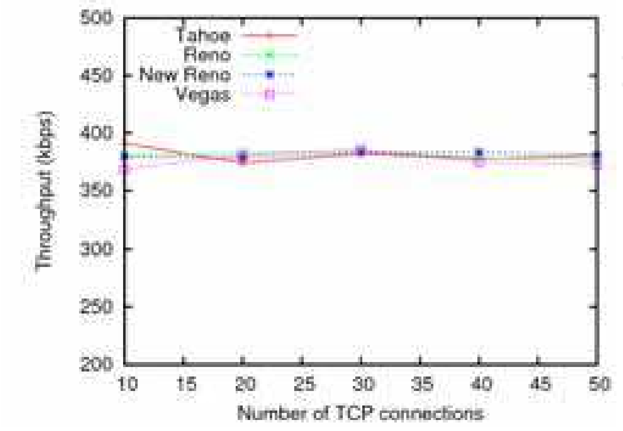
(b) DSR



(c) AODV



(d) AOMDV



(e) OLSR

Fig. 11. Throughput measurement in the random topology

5.3.3 Throughput measurement

The TCP variants over DSDV achieve a higher throughput by a factor of almost 1.5 on average compared to others as shown in Fig. 11(a). The better stability of throughput for the TCP variants could be encountered in proactive routing protocols DSDV and OLSR (Fig. 11(e)). When the number of nodes increases, the possibility of congestion and the contention at the MAC layer increase in the network. However, when the routing layer protocols receive the collision reports from the link layer, they re-discover routes by sending the broadcast messages throughout the network. Therefore, in Fig. 11(c), AODV suffers a lower throughput if compared to others. Another thing is that DSR suffers the instability throughput for all TCP variants because when the node density and the number of connections increase, the stale route problem of DSR comes active and makes the performance worse (Fig. 11(b)).

6. Conclusion

In this chapter, we analyze the performance of TCP variants across ad hoc routing protocols in static and mobile ad hoc environments. The performance of TCP variants vary depending on the routing protocols, their core mechanisms and background changes, such as the node mobility, node speed, pause time and number of tcp connections and network topologies. In the chain topology, all of the TCP variants achieve a significantly lower delay over AODV routing protocol in both environments. Moreover, AODV provides a higher throughput for all TCP variants, especially for Vegas in both environments. One interesting thing is that AODV always achieves a lower delay, it suffers a higher delay than others in the grid topology. In the grid topology, although TCP variants have the lowest delay over DSDV in both environments, in the random topology, TCP variants incur a lower packet losses over DSR and OLSR, and encounter a lower delay over DSDV. On the other hand, DSDV and OLSR provide the highest data transfer rate (i.e. throughput) for all TCP variants in random topology. Among all TCP variants, Vegas is the best transport protocol and performs better than others in most situations.

7. Acknowledgement

This work is supported in part by University of Malaya Research Grand (UMRG) under grant RG024/ 09ICT.

8. References

BonnMotion: a Mobility Scenario Generation and Analysis Tool (2009). Available from: http:/ / net.cs.unibonn.de/ fileadmin/ ag/ martini/ projekte/ BonnMotion/ src/ Bonn Motion_Docu.pdf.

Ahuja, A.; Agarwal, S.; Singh, J. P. & Shorey, R. (2000). Performance of TCP over Different Routing Protocols in Mobile Ad Hoc Networks, *IEEE 51st Vehicular Technology Conference*, pp. 2315-2319, 0-7803-5721-3, Tokyo, May 2000, Japan.

Allman, M. (1999). TCP Congestion Control, *Request for comment 2581.*

Anastasi, G.; Ancillotti, E.; Conti, M. & Passarella, A. (2007). Experimental Analysis of TCP Performance in Static Multi-hop Ad Hoc Networks, In: *Multi-hop Ad Hoc Networks from Theory to Reality*, Conti, M.; Crowcroft, J. & Passarella, A. (Ed.), page number (97-114), Nova Science, 1-60021-605-6, New York.

Boppana, R. & Konduru, S. (2001). An Adaptive Distance Vector Routing Algorithm for Mobile Ad Hoc Networks, *IEEE Infocom*, pp. 1753-1762, 0-7803-7016-3, Anchorage, April 2001, Alaska.

Brakno, L. S.; O'Malley, S. W. & Peterson, L. L. (1994). TCP Vegas: new techniques for congestion detection and avoidance, *ACM SIGCOMM Computer Communication Review*, Vol. 24, No. 4, (October 1994) page number (24-35), 0146-4833.

Camp, T., Boleng, J. & Davies, V. (2002). A survey of mobility models for ad hoc network research, *Wireless Communications and Mobile Computing Special Issue on Mobile Ad Hoc Networking: Research, Trends and Applications,* Vol. 2, No. 5, (August 2002) page number (483-502), 1530-8669.

Chandran, K.; Raghunathan, S.; Venkatesan, S. & Prakash, R. (2001). A Feedback Based Scheme for Improving TCP Performance in Ad Hoc Wireless Networks, *IEEE Personal Communication Magazine, Special Issue on Ad Hoc Networks*, Vol. 8, No. 6, (August 2001) page number (34-39), 1070-9916.

Clausen, T. & Jacquet, P. (2003). Optimized Link State Routing Protocol (OLSR), *Request for Comments 3626*.

Dube, R.; Rais, C. D.; Wang, K-Y. & Tripathi, S. K. (1997). Signal Stability-based Adaptive (SSA) Routing for Ad Hoc Mobile Networks. *IEEE Personal Communications Magazine*, Vol. 4, No. 1, (February 1997) page number (36-45), 1070-9916.

Dyer, T. D. & Boppana, R. V. (2001). A Comparison of TCP Performance over Three Routing Protocols for Mobile Ad Hoc Networks, *ACM Symposium on Mobile Ad Hoc Networking & Computing*, pp. 56-66, 1-58113-428-2, Long Beach, October 2001, ACM, California.

El-Sayed, H. M. (2005). Performance evaluation of TCP in mobile ad hoc networks, *The Second International Conference on Innovations in Information Technology,* September 2005.

Floyd, S. & Henderson, T. (1999). The NewReno Modification to TCP's Fast Recovery Algorithm, *Request for Comments 2582*.

Gerla, M.; Sanadidi, M. Y.; Zanella, R. W.; Casetti, A. & Mascolo, S. (2002). TCP Westwood: congestion window control using bandwidth estimation. *IEEE Global Telecommunications Conference*, pp. 1698-1702, 0-7803-7206-9, San Antonio, August 2002, IEEE Computer Society,TX.

Gupta, A.; Wormsbecker, I. & Williamson, C. (2004). Experimental Evaluation of TCP Performance in Multi-hop Wireless Ad Hoc Networks, *Proceedings of IEEE Annual Internation Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunications Systems*, pp. 3-11, 0-7695-2251-3, Volendam, October 2004, IEEE Computer Society, The Netherlands.

Holland, G. & Vaidya, N. (2002). Analysis of TCP Performance over Mobile Ad Hoc Networks, *Wireless Networks,* Vol. 8, No. 2/ 3 (March 2002) page number (275-288), 1002-0038.

Johnson, D.; Hu, Y. & Maltz, D. (2007). The Dynamic Source Routing Protocol (DSR) for Mobile Ad Hoc Networks for IPv4, *Request for comment 4728*.

Kawadia, V. & Kumar, P. (2005). Experimental investigation into TCP Performance over Wireless Multihop Networks, *SIGCOMM Workshops*, pp. 22-25, 1-59593-026-4, Philadelphia, August 2005, ACM, USA.

Kim, D.; Bae, H. & Song, J. (2005). Analysis of the Interaction between TCP Variants and Routing Protocols in MANETs, *Proceedings of the IEEE International Conference on Parallel Processing Workshops*, pp. 380-386, 0-7695-2381-1, University of Oslo, June 2005, IEEE Computer Society, Norway.

Lim, H.; Xu, K. & Gerla, M. (2003). TCP performance over multipath routing in mobile ad hoc networks, *IEEE International Conference on Communication*, pp. 1064-1068, 0-7803-7802-4, Anchorage, May 2003, IEEE Computer Society, Alaska.

Marina, M. K. & Das, S. R. (2006). Ad hoc on-demand multipath distance vector routing, *Wireless Communications and Mobile Computing,* Vol. 6, No. 7, (November 2006) page number (969-988), 1530-8669.

Mathis, M. & Mahdavi, J. (1996). TCP Selective Acknowledgement Options, *Request for comment 2018*.

McCanne, S. & Floyd, S. *VINT Group, Network Simulator Ns-2*. Source code: http://www.isi.edu/nsnam/ns.

Mondal, S. A. & Luqman, F. B. (2007). Improving TCP performance over wired-wireless networks, *Computer Networks, Vol.* 51, No. 13, (September 2007), page number (3799-3811), 1389-1286.

Oo, M. Z. & Othman, M. (2010). Performance Comparisons of AOMDV and OLSR Routing Protocols for Mobile Ad Hoc Network, *2010 Second International Conference on Computer Engineering and Applications*, pp. 129-133, 978-0-7695-3982-9, Bali Island, March 2010, Indonesia.

Osipov, E. & Tschudin, C. (2006). Evaluating the Effect of Ad Hoc Routing on TCP Performance in IEEE 802.11 Based MANETs, In: *Next Generation Teletraffic and Wired/Wireless Advanced Networking*, Koucheryacy, Y.; Harju, J. & Lversen, V. B. (Ed.), page number (298-312), Springer Berlin, 978-3-540-34429-2, Heidelberg.

Perkins, C.; Belding-Royer, E. & Das., S. (2003). Ad hoc on-demand distance vector routing (AODV), *Request for Comments 3561*.

Perkins, C. E. & Watson, T. J. (1994). Highly dynamic destination sequenced distance vector routing (DSDV) for mobile computers, *ACM SIGCOMM Computer Communication Review*, Vol. 24, No. 4, (October 1994) page number (234-244), 0146-4833.

Postel, J. (1981). Transmission Control Protocol (TCP), *Request for comment 793*.

Rakabawy, E. S.; Lindemann, C. & Vernon, M. (2005). Improving TCP Performance for Multihop Wireless Networks, *IEEE International Conference on Dependable Systems and Networks*, pp. 684-693, 0-7695-2282-3, Yokohama, June 2005, IEEE Computer Society, Japan.

Sakib, A. M. (2009). Improving performance of TCP over mobile wireless networks, *Wireless Networks,* Vol. 15, No. 3, (April 2009) page number (331-340), 1002-0038.

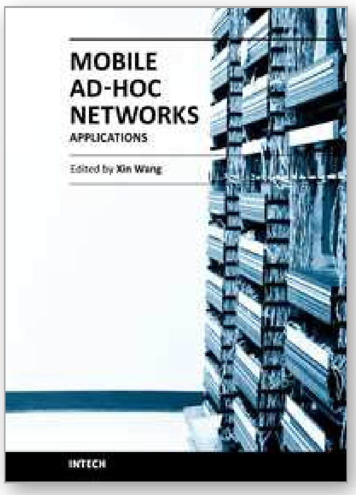
Stevens, W. (1997). TCP Slow Start, Congestion Avoidance, Fast Retransmit, *Request for comment 2001*.

Tseng, Y.-C.; Ni, S.-Y.; Chen, Y.-S. & Sheu, J.-P. (2002). The broadcast storm problem in a mobile ad hoc network. *Wireless Networks,* Vol. 8, No. 2/ 3, (March-May 2002) page number (153-167), 1002-0038.

Xu, S. & Saadawi, T. (2000). Performance Evaluation of TCP Algorithms in Multi-hop Wireless Packet Networks, *Wireless Communications and Mobile Computing*, Vol. 2, No. 1, (December 2001) page number (85-100), 1530-8669.

IntechOpen

IntechOpen



Mobile Ad-Hoc Networks: Applications
Edited by Prof. Xin Wang

ISBN 978-953-307-416-0
Hard cover, 514 pages
Publisher InTech
Published online 30, January, 2011
Published in print edition January, 2011

Being infrastructure-less and without central administration control, wireless ad-hoc networking is playing a more and more important role in extending the coverage of traditional wireless infrastructure (cellular networks, wireless LAN, etc). This book includes state-of the-art techniques and solutions for wireless ad-hoc networks. It focuses on the following topics in ad-hoc networks: vehicular ad-hoc networks, security and caching, TCP in ad-hoc networks and emerging applications. It is targeted to provide network engineers and researchers with design guidelines for large scale wireless ad hoc networks.

How to reference
In order to correctly reference this scholarly work, feel free to copy and paste the following:

May Zin Oo and Mazliza Othman (2011). The Effect of Packet Losses and Delay on TCP Traffic over Wireless Ad Hoc Networks, Mobile Ad-Hoc Networks: Applications, Prof. Xin Wang (Ed.), ISBN: 978-953-307-416-0, InTech, Available from: http://www.intechopen.com/books/mobile-ad-hoc-networks-applications/the-effect-of-packet-losses-and-delay-on-tcp-traffic-over-wireless-ad-hoc-networks

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

© 2011 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution-NonCommercial-ShareAlike-3.0 License, which permits use, distribution and reproduction for non-commercial purposes, provided the original is properly cited and derivative works building on this content are distributed under the same license.

IntechOpen

IntechOpen